

Taming uPortal: Managing Complexity, Multiple Environments, & Upgrades

Drew Wills & Cris Holdorph

Jasig Conference Denver, May 22nd, 2011

© Copyright Unicon, Inc., 2006. This work is the intellectual property of Unicon, Inc. Permission is granted for this material to be shared for non-commercial, educational purposes, provided that this copyright statement appears on the reproduced materials and notice is given that the copying is by permission of Unicon, Inc. To disseminate otherwise or to republish requires written permission from Unicon, Inc.

Introductions

Please share a bit about yourself:

- Name & Institution
- Are you running uPortal? What version?
- When was the last time you upgraded the portal framework?
- Are you running any Jasig portlets? Which ones?
- What portion of your time is spent working on the portal?



Intended Audience

- You are...
 - A Java developer, UX developer, system administrator, or similar IT professional
 - Responsible for implementing & maintaining a uPortal-based campus portal (or could be soon)
- You want to...
 - Keep up with the latest enhancements & bug fixes from Jasig
 - Skillfully manage separate *local*, *dev*, *test*, and *production* environments, seamlessly migrating content & settings between them
 - Plan for the future, putting yourself in solid position to implement major version upgrades when they arrive

Inspiration

- Drew Wills wrote an article of the same name on the Jasig wiki
- Many of the recommendations are common; the less common ones were developed in conjunction with **OHIO University & Oakland University**
- Feel free to bring additional questions to Drew at this conference

Jasig Wiki Article



Taming uPortal -- Multi-Project, Multi-Environment

Builds

Added by [Andrew Wills](#), last edited by [Andrew Wills](#) on Oct 08, 2010 ([view change](#)) [show comment](#)

For all its excellent capabilities, as a software system uPortal can be a difficult beast to work with. In its role as a Portal framework targeting the JSR-168 Portlet Specification (soon to be JSR-286, Portlet 2.0), it's very nature is to combine with and aggregate independent software projects. It is also an integration platform: it commonly requires passwords, URLs, driver classes, and diverse configuration settings to communicate properly with external systems.

These responsibilities add complexity to the portal as a software project. It's challenging to manage the relationships between the uPortal software and it's dependent projects, especially Java Portlet applications, but also non-portlet components that form a deployment unit with the portal like Apache or Tomcat. The portal, moreover, commonly expresses itself in multiple *environments*: **production**, **test**, **development**, **local**, *etc.* These environments usually require different data, configuration, integration settings.

This page describes some coordinated practices designed to tame these complexities. These solutions are not the only solutions, and may not be suitable for everyone. All options have advantages & drawbacks.

Jasig Source Code

The first order of business is obtaining source code – uPortal, community portlets, perhaps others – from Jasig. Jasig projects are *free open source software* (FOSS) and can be easily downloaded by anyone. In fact

Monday Session

A fresh Look at Building & Deploying uPortal

- Drew Wills, Unicon
- Bruce Tong, OHIO University



- **Monday May 23rd, 4:45 PM**
- Covers the use of these techniques at OHIO University
- Come if you want to review this material, choose something else if you don't, tell a colleague if you think s/he should hear it

1. What's the problem?
2. Managing Patch Releases
3. Aggregated Builds
4. Managing Multiple Portal Environments
5. Larger Upgrades

What's the Problem?

*Some factors that make
managing a portal challenging*

Why It's Hard

- uPortal isn't a software package you simply download and turn on
- For it to have any value, you have to *make it your own*
- But the lines between what adopters should customize and what Jasig should maintain & enhance are still pretty fuzzy*

**Though the situation has improved quite a bit in recent years*

Business as Usual

- In reality, most uPortal adopters make some important customizations in areas that are *genuinely likely to evolve*, such as...
 - structure or theme XSL
 - application XML (CPDs, web flows, *etc.*)
 - portlet markup
 - JavaScript,
 - Java Code
 - *or all of the above*
- Maintaining these changes in the face of on-going Jasig development is intimidating

Don't Stand Still

- But it's not a good idea simply to ignore the work in Jasig
- There's a lot going on there
- Odds are good that there are fixes for bugs you've experienced, and new features that you've wished for in the past
- *For example...*

3.2.1 Release Notes – Mar 2010

Release Notes - uPortal - Version 3.2.1 - HTML format

Configure Release Notes

Bug

- [UP-2591] - hidden content displays when portlet is dragged
- [UP-2599] - Accessibility - skip navigation links are broken
- [UP-2632] - local hsqldb stopped when tomcat running uPortal is shutdown
- [UP-2639] - Source JARs should be published with artifacts
- [UP-2645] - maven.settings should be passed to mvn executable
- [UP-2647] - Portlet rendering issue, contents of one portlet are rendered in another portlet on the same page.
- [UP-2648] - Mist theme class missing from uportal3 skin.xml
- [UP-2651] - Groups selector subflow does not properly toggle class on selected group
- [UP-2652] - Upgrade jQuery UI library to patched datepicker version

New Feature

- [UP-2650] - Add ability to suppress portlets from mobile view

Task

- [UP-2646] - reset-user-layout.channel should not hard-code the "portletApplicationId"

3.2.2 Release Notes – Jul 2010

Release Notes - uPortal - Version 3.2.2 - HTML format

Configure Release Notes

Sub-task

- [UP-2315] - Move uportal-impl/src/main/resources/properties/db/entities out of the uportal-impl module
- [UP-2384] - NPE uportal-impl/src/main/java/org/jasig/portal/car/DescriptorHandler.java
- [UP-2387] - NPE uportal-impl/src/main/java/org/jasig/portal/groups/filesystem/FileSystemGroupStoreFactory.java
- [UP-2665] - Update channel export to use the ChannelDefinition API, fix regression

Bug

- [UP-706] - CWebProxy appends parameters after #
- [UP-1212] - Changing the user locale does not change the session locale
- [UP-1599] - Exception Closing Oracle DB Connection under WebSpere 6.1
- [UP-1624] - CLONE -WEBPROXY channel only passes IPERSON parameters on the initial GET call
- [UP-1884] - NPE in RDBMUserIdentityStore.removePortalUID
- [UP-2382] - Sitemap - fifth tab does not properly wrap to the next row
- [UP-2565] - Removing published portlets from portlet manager causes uportal to crash with the generic error
- [UP-2608] - coal and uportal3 skin render problem
- [UP-2612] - can't use "add content" chooser with coal skin
- [UP-2614] - Fragment Administration portlet fails on coal skin
- [UP-2615] - The Exit Fragment Administration portlet css need to be updated for the ivy and coal skins

3.2.2 Release Notes – Jul 2010

- [UP-2627] - Prefix must resolve to a namespace error on first startup
- [UP-2629] - Maximised mode portlet controls show images in background
- [UP-2630] - ORA-01000: maximum open cursors exceeded
- [UP-2635] - Administration Portlet gives error if no group selected
- [UP-2643] - SystemId not set for source in XSLT (affect xsl:import)
- [UP-2668] - .car archive are never redeployed even if processIf tags says it should be
- [UP-2669] - Unable to Select the Everyone Group when Assigning Groups to Portlets
- [UP-2672] - Users can't subscribe to just-created channels
- [UP-2673] - Errors in CSpringPortletAdaptor.cpd
- [UP-2681] - Can not select Category or Group in Portlet Manager with Firefox 3.6.2 or Internet Explorer 8
- [UP-2682] - Edit Portlet Mode control does not work in uPortal 3.2.1
- [UP-2685] - Bug in init.crn XML file in SmartLdapGroupsStore prevents it from properly analyzing groups in LDAP
- [UP-2687] - SQL statement with incorrect number of parameters
- [UP-2690] - Administrators do not have subscribe permissions on non-published content
- [UP-2696] - Can not use "Groups Administration" portlet in uPortal 3.2.1 despite the option being available to users
- [UP-2697] - CONFIG mode is inaccessible
- [UP-2698] - Bug in CacheSecurityContext prevents it from working with any "real" SecurityContext except SimpleSecurityContext
- [UP-2700] - Encoding pb "add content" - XmlView ChannelList - ISO-8859-1
- [UP-2703] - Delete Fluid version variable from public scope
- [UP-2704] - unicon-news does not work in uPortal 3.2.x
- [UP-2705] - UP_VERSIONS is not updated for 3.2.1

3.2.2 Release Notes – Jul 2010

- [UP-2712] - Can only search once in Portal Admin Portlet - Register New Portlet - Select Categories and Select People and Groups
- [UP-2714] - In portlet.xml of uPortal webapp, RegisterPortalPortlet has display-name "Attribute Swapper Portlet"
- [UP-2731] - After deleting expired portlets, uPortal fails to render
- [UP-2732] - WebProxyPortlet Rich Portlet Config fails out of box
- [UP-2971] - CLONE - Encoding pb "add content" - XmlView ChannelList - ISO-8859-1

Improvement

- [UP-2617] - List portlet webapps in alphabetical order in Portlet Administration register new portlet dropdown
- [UP-2655] - Upgrade to Fluid Infusion 1.2 for mobile theme CSS
- [UP-2688] - SmartLdapGroupStore improperly looks for membership relationships for non-IPerson entities
- [UP-2694] - Add css classes for portlet form ui to denote disabled states
- [UP-2710] - Remove the static block from SmartLdapGroupStore
- [UP-2711] - Show dynamic title option for portlet publishing type
- [UP-2722] - CAS Proxy Test Portlet's service URL should be configurable

3.2.2 Release Notes – Jul 2010

New Feature

- [UP-139] - saving minimized channel state
- [UP-2600] - Accessibility - add a high-contrast skin
- [UP-2640] - Administration Portlet warns " Please enter an fname" when i forgot entering the "channel Name" in practice
- [UP-2692] - Enhance SmartLdap groups to optionally resolve member groups existing outside the normal baseDn
- [UP-2706] - Allow RSS reader to use proxy server

Task

- [UP-2654] - Update libraries
- [UP-2752] - Get the Maven release plugin working
- [UP-2754] - Upgrade Libraries for 3.2.2

3.2.3 Release Notes – Oct 2010

Release Notes - uPortal - Version 3.2.3 - HTML format

Configure Release Notes

Bug

- [UP-2765] - Sitemap tab links make you "stuck" on the selected tab
- [UP-2768] - Fix List<IPortletPreference> persistence to reduce DB churn
- [UP-2781] - uPortal Error When Changing Skin
- [UP-2795] - Unicon News portlet pointed at broken RSS feed
- [UP-2797] - Portal Administrator lacks permission to access Password Management channel
- [UP-2816] - JPA Channel Definition code is poorly cached
- [UP-2828] - JavaScript Error - 'script.parentNode is null or not an object' - upgrade Fluid to 1.2.1 or 1.1.3
- [UP-2829] - Portlet output cache not invalidation when chrom control clicked
- [UP-2830] - QueueingEventHandler dispose throws exceptions if there are queued events
- [UP-2832] - Don't remove guest layout from cache
- [UP-2833] - Gracefully handle poorly configured dlm.xml
- [UP-2834] - Portlet FName URLs ignore additional parameters
- [UP-2835] - Framework webflow portlets don't minimize
- [UP-2839] - Lifecycle options don't display for portlets with no categories
- [UP-2840] - Logging of XML in StaticRenderingPipeline uses inconsistent checks

3.2.3 Release Notes – Oct 2010

Improvement

- [UP-2258] - Add name to iframe channel
- [UP-2762] - Handle orphaned/corrupt channel SUBSCRIBE permissions more gracefully in export-channel.crn (Import/Export)
- [UP-2777] - Enhance SmartLdap to refresh group relationships without a server restart
- [UP-2804] - Improve uPortal 3.x performance by avoiding the persistence of unused portlet entities.
- [UP-2817] - Check JDK Version in quickstart ant script
- [UP-2838] - Lengthen default channel rendering timeout from 500ms to 5,000ms
- [UP-2841] - Allow CDATA to be used in .channel files

3.2.4 Release Notes – Oct 2010

Release Notes - uPortal - Version 3.2.4 - HTML format

Configure Release Notes

Bug

- [UP-2844] - Error exporting layouts with DLM node references with some JDBC drivers
- [UP-2854] - Portlets fail when maximized

Improvement

- [UP-2856] - Update Resource Aggregator

3.2.5 Release Notes – ???

UP-2895	Remove unused channels from default layouts
UP-2894	Expanded header and footer content from DLM fragments
UP-2911	Portlet preferences don't show in portlet manager review screen
UP-2900	Bookmarks portlet doesn't start on uPortal 3.2.2
UP-2928	Portlet Entity Persistence is not Thread Safe
UP-2899	Export Portlet Fails
UP-2927	Provide a 'Reset My Layout' Portlet
UP-2906	WebProxyPortlet needs to filter from uportal-impl/target.../rdm.properties now that filtering is in place
UP-2674	Complete uPortal NOTICE file
UP-2622	Fragment owners should have their own template layout
UP-2805	uPortal icon link does not take user to Home tab
UP-2889	NPE in PersonImpl that has not been fully initialized
UP-2857	Portlets containing namespace tag fail on first request in mobile theme
UP-2881	Resource Aggregation Toggle Broken

3.2.5 Release Notes – ???

UP-2882	Toggleing Aggregation doesn't clear cache
UP-2858	upgrade quartz dependency to version 1.8.4
UP-2957	Add a framework portlet for auditing the current DLM fragments
UP-2958	Enhance the DLM fragment audit portlet to show included portlets as well
UP-2970	Provide portlets with reports on which portlets have been added by users and how many times
UP-2980	Browser reload in CONFIG mode crashes Portlet Manager portlet
UP-2619	Receiving "unknown error occurred" JS messages trying to add portlets to locked-down DLM fragments
UP-2228	Add "revert to default layout" button to page layout dialog
UP-2948	Enhance GaP (groups) to be more resilient to abrupt loss of group nodes in external systems
UP-2785	"Page needs a session and none is available" error in catalina.out
UP-2887	Add support for externally hosted CSS files
UP-2886	escaping values in portlet-admin
UP-2890	Enhance SmartLdapGroupStore to detect and ignore circular group references
UP-2476	wrong suffix in import/export online tool for fragment owners
UP-2671	Add Tab to Fragment - Step Count Typo and Mystery Radio Button

Does This Portlet Useful?

Portlet Administration Portlet

[← Return to dashboard](#)

[+ Add to my layout](#)

Most Frequently Added

Previous days from inclusive

[< previous](#) [1](#) [2](#) [3](#) [4](#) [5](#) [6](#) [7](#) [8](#) [9](#) [10](#) [11](#) [12 \(last\)](#) [next >](#) 1-10 of 114 items per page

Title	# Times
Blackboard Courses	76
My OHIO Student Center	64
Academic Calendar	57
Blackboard Announcements	37
Outlook Live Email	36
Student Services (Registrar)	30
Registrar Information	29
Financial Aid Information	28
Alice Online Catalog	25
Athens Campus Weather	24

Done

What About This One?

Audit DLM Fragments ⌵ ✕

The following DLM fragments are currently active in the portal:

Guests

Owner: guest-lo Precedence: 500

Audience:

- (GUEST)

Portlets:

- Welcome to uPortal
- Logging in
- uPortal Links
- Please Register

Home

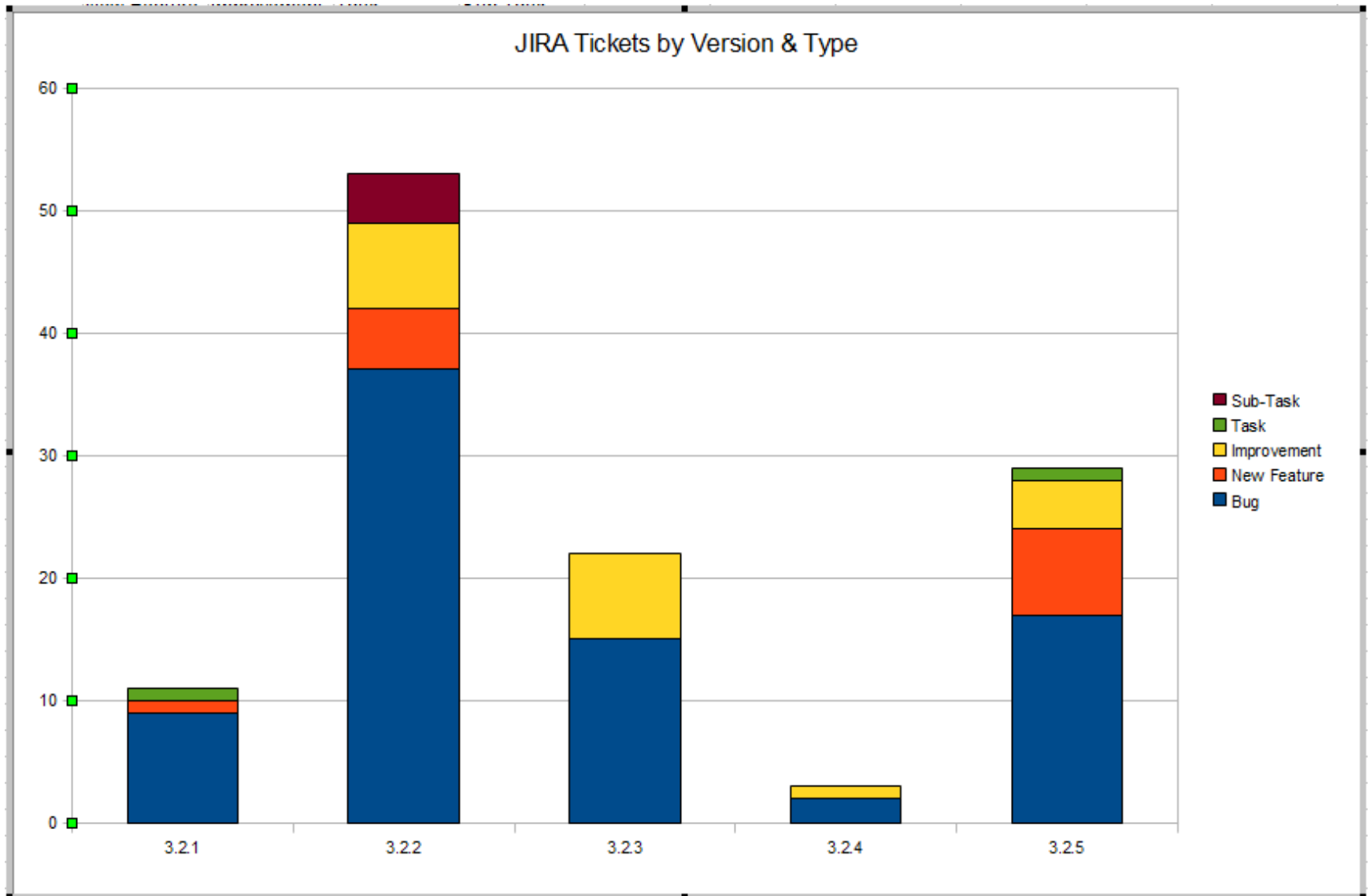
Owner: home-lo Precedence: 400

Audience:

- ((MEMBER OF 'Everyone' OR ANY DESCENDANT GROUP) && !(MEMBER OF 'Guests' OR ANY DESCENDANT GROUP))

Portlets:

JIRA Summary 3.2.1 - 2011/05/01



So What Are You Saying?

- A single institution probably can't compete with the pace of innovation in the project itself
- Even if it you could, why would you want to?
- Wouldn't it be better to *benefit from both your own efforts and the community contributions* continuously... at the same time?
- The practices in the next section are designed to help you do just that

Managing Local Customizations

How do I make it OK that my source and Jasig's source change independantly?

Keeping up with Jasig

- A well-designed relationship between your source code and Jasig project code can make all the difference applying incremental changes
- Consider these practices
 - Vendor branching
 - `svn:externals`
 - Maven Overlays
 - Do-It-Yourself (DIY) Overlays
- They can greatly simplify the process of pulling in new code from Jasig

Vendor Branching

- Widely adopted practice for addressing this class of problem
- Captures the difference between 2 versions of 3rd-party source code as a *delta* that can be merged, automatically, with your customizations and into your project
- Each new version of 3rd-party source is called a *vendor drop*

Vendor Branching: SVN Example

- Add 3rd-party source to a special area called vendor/

```
>svn import rel-3-2-0-GA https://svn.my.edu/vendor/uPortal/current/ \  
    -m "Initial uP 3.2.0 vendor drop"  
  
>svn copy https://svn.my.edu/vendor/uPortal/current/ \  
    https://svn.my.edu/vendor/uPortal/3.2.0/
```

- Copy the vendor drop to a tag named for the current version

Vendor Branching: SVN Example

- Copy the tag* to a location in your project and check it out

```
>svn copy https://svn.my.edu/vendor/uPortal/3.2.0/ \  
          https://svn.my.edu/trunk/ \  
          -m "Folding uP 3.2.0 into the project"  
  
>svn co https://svn.my.edu/trunk/  
  
>cd trunk
```

- Make & commit local customizations to the 3rd-party source in your checkout

**or perhaps current/, if it seems clearer*

Vendor Branching: SVN Example

- When a new version is available, replace the contents of `current/` with the latest sources

```
>svn co https://svn.my.edu/vendor/uPortal/current/  
>copy rel-3-2-1-GA current  
>cd current  
>svn commit -m "Newer uP 3.2.1 vendor drop"
```

- **Warning!** This is a simple example – no files, added, deleted, or moved

Vendor Branching: SVN Example

- Merge the *difference* between the old version and the new version into your project

```
>cd trunk  
  
>svn merge https://svn.my.edu/vendor/uPortal/3.2.0/ \  
           https://svn.my.edu/vendor/uPortal/current/  
  
# Resolve conflicts between their code & your code  
  
>svn commit -m "Updating uP to version 3.2.1"
```


svn_load_dirs.pl

- Okay, but what happens with a real project?
- The previous example won't correctly handle...
 - Added files
 - Deleted files
 - Moved files
- Subversion provides a perl script that can help you manage these complexities

```
>svn_load_dirs.pl [-t 3.2.0] https://svn.my.edu/vendor/uPortal \  
current rel-3-2-1-GA|
```

Vendor Branching Pros & Cons

- **Pros:**
 - Blend changes from Jasig & local changes with automated merge tools
 - Industry-standard, well documented practice
- **Cons:**
 - Not particularly easy or quick
 - Must be done correctly from the beginning; make a mistake at any point and you loose the advantage
 - You still have to resolve conflicts by hand
 - Local changes to files that move are lost
 - `svn_load_dirs.pl` does not support properties

svn:externals

- **Externals Definitions** is an alternative to vendor branching
- Compose a working copy from separate but aggregated checkouts
- It even works across (SVN) repositories
- Allows you to *develop on the real project* and commit patches directly!

.externals File

- Using an .externals file to track changes is a popular convention

```
uPortal -r 23430 https://source.jasig.org/uPortal/branches/rel-3-2-patches/  
email-preview -r 23391 https://source.jasig.org/portlets/email-preview/trunk/  
FeedbackPortlet -r 23025 https://source.jasig.org/sandbox/FeedbackPortlet/trunk/  
AnnouncementsPortlet -r 22710 https://source.jasig.org/portlets/AnnouncementsPortlet/trunk/  
CalendarPortlet -r 23325 https://source.jasig.org/portlets/CalendarPortlet/trunk/  
JasigWidgetPortlets -r 20743 https://source.jasig.org/sandbox/JasigWidgetPortlets/trunk/  
SimpleContent -r 22815 https://source.jasig.org/portlets/SimpleContentPortlet/trunk/  
TabbedSearch -r 22623 https://source.jasig.org/portlets/TabbedSearchPortlet/trunk/  
WeatherPortlet -r 23340 https://source.jasig.org/portlets/WeatherPortlet/trunk/
```

- It's easier to make changes to a text file
- You can view it in a web browser
- Be certain to **specify a revision number** for each external item

svn:externals Setup

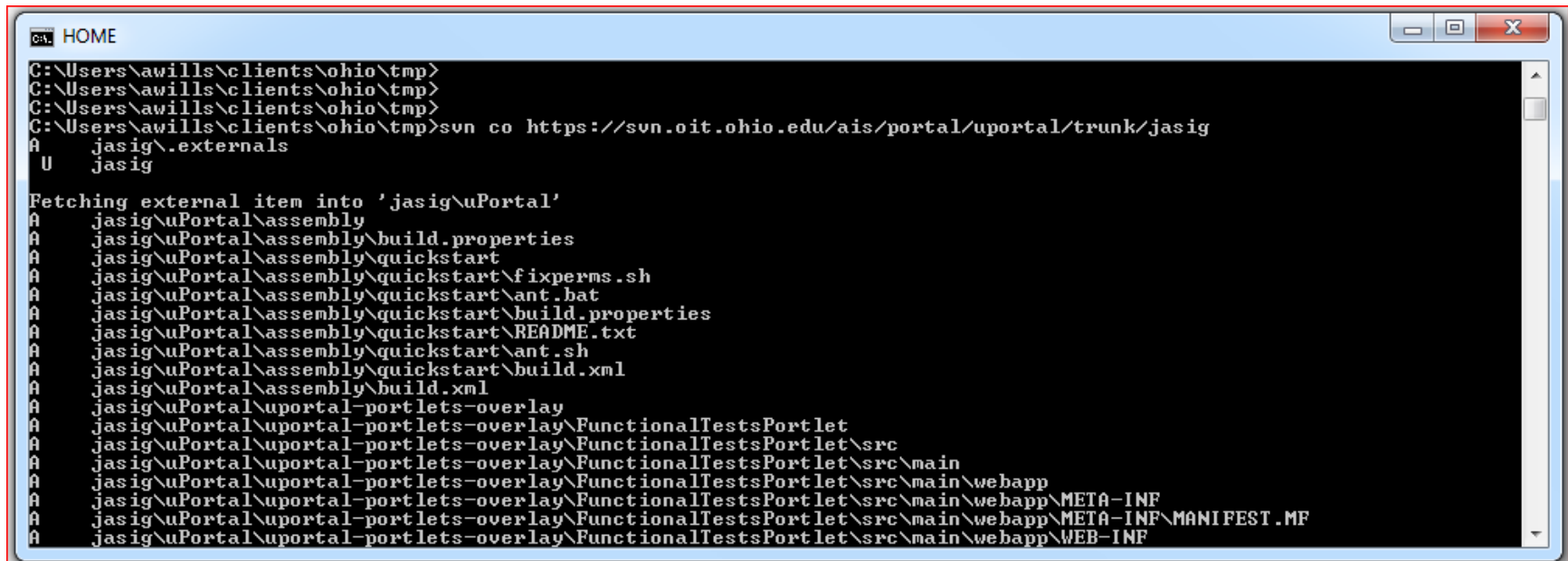
- Create the file first, then use `svn propset -F`

```
>svn add .externals
>svn propset svn:externals -F .externals .
>svn commit -N . .externals -m "Incorporating external checkouts"
```

- Commit the property & the file at the same time, atomically
- To change the version of an external dependency, just edit the file and repeat the process

svn:externals Checkout

- When you checkout your project, Subversion automatically includes external directories where you place them



```
HOME
C:\Users\awills\clients\ohio\tmp>
C:\Users\awills\clients\ohio\tmp>
C:\Users\awills\clients\ohio\tmp>
C:\Users\awills\clients\ohio\tmp>svn co https://svn.oit.ohio.edu/ais/portal/uportal/trunk/jasig
A   jasig\.externals
U   jasig

Fetching external item into 'jasig\uPortal'
A   jasig\uPortal\assembly
A   jasig\uPortal\assembly\build.properties
A   jasig\uPortal\assembly\quickstart
A   jasig\uPortal\assembly\quickstart\fixperms.sh
A   jasig\uPortal\assembly\quickstart\ant.bat
A   jasig\uPortal\assembly\quickstart\build.properties
A   jasig\uPortal\assembly\quickstart\README.txt
A   jasig\uPortal\assembly\quickstart\ant.sh
A   jasig\uPortal\assembly\quickstart\build.xml
A   jasig\uPortal\assembly\build.xml
A   jasig\uPortal\uportal-portlets-overlay
A   jasig\uPortal\uportal-portlets-overlay\FunctionalTestsPortlet
A   jasig\uPortal\uportal-portlets-overlay\FunctionalTestsPortlet\src
A   jasig\uPortal\uportal-portlets-overlay\FunctionalTestsPortlet\src\main
A   jasig\uPortal\uportal-portlets-overlay\FunctionalTestsPortlet\src\main\webapp
A   jasig\uPortal\uportal-portlets-overlay\FunctionalTestsPortlet\src\main\webapp\META-INF
A   jasig\uPortal\uportal-portlets-overlay\FunctionalTestsPortlet\src\main\webapp\META-INF\MANIFEST.MF
A   jasig\uPortal\uportal-portlets-overlay\FunctionalTestsPortlet\src\main\webapp\WEB-INF
```

- If you change the revision, Subversion automatically changes the external directory on update

svn:externals & Local Customizations

- Unlike vendor branching, `svn:externals` will not combine your customizations with Jasig code automatically
- Consider *overlaying* local customizations on top of Jasig source
- We will look at 2 approaches:
 - Maven Overlays
 - DIY

Maven Overlays

- Feature of the Maven War Plugin
- Works with `<packaging>war</packaging>` projects
- Simple to set up: include another war as a dependency of your war

```
<artifactId>WeatherPortlet</artifactId>
<packaging>war</packaging>

<name>Weather Portlet</name>
<description>Overlay on Weather Portlet.</description>

<dependencies>
  <dependency>
    <groupId>org.jasig.portlet</groupId>
    <artifactId>WeatherPortlet</artifactId>
    <version>${WeatherPortlet.version}</version>
    <type>war</type>
  </dependency>
</dependencies>
```


Maven Overlays (cont.)

- Files in your project replace files of the same name in the dependency
- You can also tweak overlay behavior
 - e.g. include/exclude files specifically

```
<overlay>
  <groupId>org.jasig.resourceserver</groupId>
  <artifactId>resource-server-content</artifactId>
  <includes>
    <include>rs/fluid/1.1.3</include>
    <include>rs/jquery/1.3.2</include>
    <include>rs/jqueryui/1.7.2</include>
    <include>rs/famfamfam/silk/1.3/arrow_refresh_small.png</include>
    <include>rs/famfamfam/silk/1.3/attach.png</include>
    <include>rs/famfamfam/silk/1.3/delete.png</include>
    <include>rs/famfamfam/silk/1.3/door_out.png</include>
    <include>rs/famfamfam/silk/1.3/email.png</include>
    <include>rs/famfamfam/silk/1.3/flag_red.png</include>
  </includes>
</overlay>
```

DIY Overlays

- Just like Maven Overlays, but without Maven's help (e.g. dependency management)
- Use a `work/` directory to combine original source files with your customizations

```
>mkdir work  
  
>copy original/uPortal work/uPortal  
  
>copy overlay/uPortal work/uPortal  
  
>cd work/uPortal  
  
>ant initportal
```

- Works with any type of project

Comparison Chart

	Vendor	externals	mvn	DIY
Combine local & Jasic code	Y	N	Y	Y
Fast, simple upgrades	N	Y	Y	Y
Merge local/Jasig changes automatically	Y	N	N	N
Commit against the real project	N	Y	N/A	N/A
Clear, well-documented practice	Y	Y	Y	N
Supports any build system & project type	Y	Y	N	Y

Break



Aggregated Builds

Building the Portal & Portlets together

Built-In Build Capabilities

- uPortal provides a blended Ant/Maven build system

```
>ant initportal  
# or...  
>ant deploy-ear  
# or...  
>ant deploy-war
```

- Jasig portlets typically provide a Maven-based build system

```
>mvn install  
# or...  
>mvn package
```

Built-In Build Capabilities (cont.)

- Especially with uPortal, there's a lot of custom logic & sophistication baked into the build
 - Maven/Ant integration
 - Database management
 - Web server deployment
 - Nested modules
 - LICENSE and NOTICE plugins
 - Unit tests and static analysis
 - Pluto
 - yuicompressor and resource-aggregator

Built-In Build Capabilities (cont.)

- These build-related settings and behaviors change frequently
- When they do, they often don't make pretty diffs
- So it usually doesn't pay to reinvent or fork the build process

“Puppet Master” Build

- Although the inner-workings of the build systems commonly change in confusing ways, the way(s) you invoke them generally don't
- So you can safely *aggregate* the builds of uPortal and related projects
- For this goal, I suggest Groovy
 - Java-based syntax
 - Platform-independent



“Puppet Master” Build Example

- Use the same installations of Ant and Maven as building the projects directly

```
1 // Use a prefix for shell output to improve clarity
2 def PFX = '[build.groovy]';
3
4 // Location of the Apache Ant executable (platform independent)
5 def ANT_HOME = System.getenv('ANT_HOME');
6 def ANT_FILENAME = System.getProperty('os.name') =~ /Windows/ ? 'ant.bat' : 'ant';
7 def ANT_EXEC = "${ANT_HOME}/bin/${ANT_FILENAME}";
8
9 // Location of the Apache Maven executable (platform independent)
10 def M2_HOME = System.getenv('M2_HOME');
11 def M2_FILENAME = System.getProperty('os.name') =~ /Windows/ ? 'mvn.bat' : 'mvn';
12 def M2_EXEC = "${M2_HOME}/bin/${M2_FILENAME}";
13
```

Build Arguments Example

- Turn portions of the build on or off using additional arguments
- **Saves considerable time** in development

```
14 // Flags to skip some stages of the full build; good for fast local builds
15 def skipClean = Boolean.valueOf(System.getProperty('build.clean.skip'));
16 def skipPortal = Boolean.valueOf(System.getProperty('build.portal.skip'));
17 def skipPortlets = Boolean.valueOf(System.getProperty('build.portlets.skip'));
18
19 // Optional Ant target to run; the default is 'deploy-ear' but 'deploy-war'
20 // will be faster if you don't need to process webapps other than uPortal
21 def antTarget = System.getProperty('build.ant.target') ?: 'deploy-ear';
```

>groovy -Dbuild.clean.skip=true build.groovy

Maven Settings Example

- Remember to pass along Maven and/or Ant flags that you might use

```
22 // Maven settings
23 def envSettings = "";
24 if (System.getProperty('maven.test.skip')) {
25     envSettings += " -Dmaven.test.skip=${System.getProperty('maven.test.skip')}";
26 }
27 if (System.getProperty('maven.offline')) {
28     envSettings += " -Dmaven.test.skip=${System.getProperty('maven.offline')}";
29 }
30
```

>groovy -Dmaven.test.skip=true build.groovy

uPortal Reset Example

- Reset the work/uPortal/ directory unless overridden by the build.skip.clean flag (above)

```
30 def ant = new AntBuilder();
31
32 // Clean & reset uPortal source code from Jasig
33 if (!skipPortal && !skipClean) {
34
35     println "${PFX} * * * * *";
36     println "${PFX} Clearing out the previous build";
37     println "${PFX} * * * * *";
38     println "${PFX}";
39
40     ant.delete(dir:'work/uPortal');
41
42     println "${PFX} * * * * *";
43     println "${PFX} Bringing in the uPortal source";
44     println "${PFX} * * * * *";
45     println "${PFX}";
46
47     ant.copy (todir:'work/uPortal') {
48         fileset(dir:'original/uPortal') {
49             exclude(name:'**/.svn')
50         }
51     };
52
53 }
```

Build uPortal Example

- Apply local customizations, build with Ant

```
56 def exitValue = 0; // Indicates success
57
58 // Apply local customizations
59 if (!skipPortal) {
60
61     println "${PFX} * * * * *";
62     println "${PFX} Overlaying local files";
63     println "${PFX} * * * * *";
64     println "${PFX}";
65
66     ant.copy (todir:'work/uPortal', overwrite:true) {
67         fileset(dir:'overlay/uPortal') {
68             exclude(name:'**/.svn')
69         }
70     };
71
72     println "${PFX} * * * * *";
73     println "${PFX} Invoking uPortal's build process to make JARs and WARs.";
74     println "${PFX} * * * * *";
75     println "${PFX}";
76
77     // Prepare the ant command
78     def cmd = "${ANT_EXEC} ${envSettings} ${antTarget}";
79
80     // Execute the command with an Ant process
81     def process = cmd.execute(null, new File('work/uPortal'));
82     process.consumeProcessOutput(System.out, System.err);
83     process.waitFor();
84     exitValue = process.exitValue();
85
86 }
```

Building Portlets Example

- Add portlets to the Puppet Master build

```
93  if (!skipPortlets) {
94
95     println "${PFX} * * * * *";
96     println "${PFX} Building & deploying additional portlets.";
97     println "${PFX} * * * * *";
98     println "${PFX}";
99
100    def targetPortlet = System.getProperty('build.target.portlet');
101    def portletsToDeploy = ((targetPortlet != null)
102        ? ALL_PORTLETS.subMap([targetPortlet])
103        : ALL_PORTLETS);
104
105    portletsToDeploy.each { portletName, deployer ->
106
107        println "${PFX} * * * * *";
108        println "${PFX} ${portletName}.";
109        println "${PFX} * * * * *";
110        println "${PFX}";
111
112        exitValue = deployer();
113
114        if (exitValue != 0) {
115            System.exit(exitValue);
116        }
117    }
118 }
119
120 }
```

Maven Overlay Portlet Example

- No need to delete/copy files in a work/ directory

```
89 def ALL_PORTLETS = [  
90     'WeatherPortlet': {  
91  
92         def wpBuild = "${M2_EXEC} ${envSettings} clean package";  
93         def wpBuildProcess = wpBuild.execute(null, new File('overlay/WeatherPortlet'));  
94         wpBuildProcess.consumeProcessOutput(System.out, System.err);  
95         wpBuildProcess.waitFor();  
96         def rslt = wpBuildProcess.exitValue();  
97         if (rslt != 0) {  
98             return rslt;  
99         }  
100  
101         def wpDeploy = "${ANT_EXEC} ${envSettings} deployPortletApp " +  
102             "-DportletApp=../../overlay/WeatherPortlet/target/WeatherPortlet.war";  
103         def wpDeployProcess = wpDeploy.execute(null, new File('work/uPortal'));  
104         wpDeployProcess.consumeProcessOutput(System.out, System.err);  
105         wpDeployProcess.waitFor();  
106         rslt = wpDeployProcess.exitValue();  
107  
108         return rslt  
109     },  
110 ],  
111
```


DIY Overlay Portlet Example

```
89 def ALL_PORTLETS = [  
90   'email-preview': {  
91     ant.delete(dir:'work/email-preview');  
92     ant.copy(todir:'work/email-preview') {  
93       fileset(dir:'original/email-preview') {  
94         exclude(name:'**/.svn')  
95       }  
96     }  
97   };  
98  
99   ant.copy (todir:'work/email-preview', overwrite:true) {  
100     fileset(dir:'overlay/email-preview') {  
101       exclude(name:'**/.svn')  
102     }  
103   };  
104  
105   def epBuild = "${M2_EXEC} ${envSettings} clean package";  
106   def epBuildProcess = epBuild.execute(null, new File('work/email-preview'));  
107   epBuildProcess.consumeProcessOutput(System.out, System.err);  
108   epBuildProcess.waitFor();  
109   def rslt = epBuildProcess.exitValue();  
110   if (rslt != 0) {  
111     return rslt;  
112   }  
113  
114   def epDeploy = "${ANT_EXEC} ${envSettings} deployPortletApp " +  
115     "-DportletApp=../../work/email-preview/target/email-preview.war";  
116   def epDeployProcess = epDeploy.execute(null, new File('work/uPortal'));  
117   epDeployProcess.consumeProcessOutput(System.out, System.err);  
118   epDeployProcess.waitFor();  
119   rslt = epDeployProcess.exitValue();  
120  
121   return rslt  
122 }  
123 ],
```

Managing Multiple Portal Environments

Maintaining distinct settings & data for
PROD, TEST, DEV, *etc.*

2 Important Techniques

This section covers two key areas for supporting multiple environments on the same codebase:

- **Configuration:** using Maven filters to manage separate databases, log settings, *etc.*
- **Data:** using Import/Export to manage common data & environment-specific data, and to move data around

Maven Filters

- Allows project files to contain values that will be supplied at build time
- These values can come from several sources:
 - The pom file (e.g. `${pom.version}`)
 - The settings file (e.g. `${settings.localRepository}`)
 - Pom `<properties>` (e.g. `${my.custom.value}`)
 - `-D` parameters (e.g. `mvn -Dfoo=bar install`)
 - **A filters file**

Filters Files

- Use Maven filters files to gather values for filters into one file

```
<build>
  <filters>
    <filter>src/main/filters/filter.properties</filter>
  </filters>
  <resources>
    <resource>
      <directory>src/main/resources</directory>
      <filtering>true</filtering>
    </resource>
  </resources>
</build>
```

- Use a different file for each environment!
- **WARNING:** Never filter binary files

Maven Filters in uPortal 4

UP-2813: *Add hooks for Maven filters to uPortal poms to support multi-environment builds*

- Use `build.properties` itself as a filters file
- Or use `build.${env}.properties` for multiple environments if you want to keep them in the same place
- Or choose your own location by specifying the `filters.file` property in `build.properties`

build.properties

```
#----- Maven Filtering -----
# You can use this file to manage environment-specific settings and supply them
# to the appropriate locations at build time. This process uses a Maven feature
# called "Filtering" (http://maven.apache.org/shared/maven-filtering/).
#
# Place filter tokens for environment-specific settings in configuration files,
# then provide values for those tokens below. NOTE: Files with filter tokens
# MUST be listed in the <includes> section for filtering in the appropriate
# pom.xml file.

# Use 'filters.file' to override the location of the properties file for
# Maven filtering. The specified location should be a RELATIVE PATH. If not
# specified, this file (build.properties or build.{env}.properties) will be used.
#filters.file=filters/prod.properties

## EXAMPLES ##

## HSQL Configuration
environment.build.hsql.port=8887

## Database Connection Settings (Uncomment the Maven Filters section in rdbm.properties)
environment.build.hibernate.connection.driver_class=org.hsqldb.jdbcDriver
environment.build.hibernate.connection.url=jdbc:hsqldb:sql://localhost:${environment.build.hsql.port}
environment.build.hibernate.connection.username=sa
environment.build.hibernate.connection.password=
environment.build.hibernate.dialect=org.hibernate.dialect.HSQLDialect

# uPortal server configuration properties
environment.build.uportal.server=localhost:8080
environment.build.uportal.protocol=http
environment.build.uportal.context=/uPortal
environment.build.uportal.email.fromAddress=portal@university.edu

# CAS server configuration properties
environment.build.cas.server=localhost:8080
environment.build.cas.protocol=http
```



build.\${env}.properties

- You can optionally manage all your build.properties files in the same location
- Name each one for its environment, *e.g.*:
 - build.**local**.properties
 - build.**dev**.properties
 - build.**test**.properties
 - build.**prod**.properties
- Invoke Ant for a named environment as follows:

```
>ant -Denv=dev clean initportal
```


build.xml

- Ant passes the location of your filters file to Maven as `-Denvironment.name`

```
<condition property="env.arg" value="-Denvironment.name=${env}.">   
  <isset property="env" />  
</condition>  
<property name="env.arg" value="-Djasig.ignore" />  
  
<condition property="filters.arg" value="-Dfilters.file=${filters.file}.">  
  <isset property="filters.file" />  
</condition>  
<property name="filters.arg" value="-Djasig.ignore" />  
  
<artifact:mvn pom="${pomDir}/pom.xml" failonerror="true" fork="true" mavenHome="${maven.home}"  
  <arg value="-s${maven.settings}" />  
  <arg value="${test.skip}" />  
  <arg value="${offline}" />  
  <arg value="${env.arg}" />   
  <arg value="${filters.arg}" />  
  <arg value="${goal}" />  
  <arg value="${goal1}" />  
  <arg value="${goal2}" />
```

pom.xml

- Maven uses the `environment.name` parameter to evaluate the location of the filters file*

```
<!--
| Environment name and filters file for environment-specific build
| settings. By default <environment.name> is blank, and the filters
| file is build.properties.
|
| You can override the default by passing a -Denvironment.name
| parameter, which Ant will do for you if you invoke Ant with
| -Denv={something}. NOTE that Ant also adds a suffixed period (.)
| character. If you want to override the default environment.name by
| invoking Maven directly, you will have to append a period yourself.
+-->
<environment.name></environment.name>
<filters.file>build.${environment.name}properties</filters.file>
```

- The default is `build.properties`

**An explicitly-specified `filters.file` property overrides `environment.name` if present*

uportal-war/pom.xml

- uportal-war filters resource files with the specified filters.file

```
<build>
  <finalName>${uportal.docbase}</finalName>
  <filters>
    <filter>${basedir}/../${filters.file}</filter>
  </filters>
  <resources>
    <resource>
      <directory>src/main/resources</directory>
      <filtering>true</filtering>
      <!--
      | Enumerate filtered files explicitly to avoid issues with other
      | config tech, e.g. Spring property placeholder
      +-->
      <includes>
        <include>properties/portal.properties</include>
        <include>properties/rdbm.properties</include>
        <include>properties/security.properties</include>
        <include>properties/contexts/userContext.xml</include>
      </includes>
    </resource>
    <!--
    | Declare a mutually exclusive resource set for non-filtered files.
    +-->
    <resource>
      <directory>src/main/resources</directory>
      <filtering>false</filtering>
      <excludes>
        <exclude>properties/portal.properties</exclude>
        <exclude>properties/rdbm.properties</exclude>
        <exclude>properties/security.properties</exclude>
        <exclude>properties/contexts/userContext.xml</exclude>
      </excludes>
    </resource>
  </resources>
</build>
```

rdbm.properties

- Use filter expressions to set database connection properties in uPortal 4

```
##### Maven Filtering
##### Use these settings to define DB connection settings in build.properties,
##### build.{env}.properties, or a filters file
hibernate.connection.driver_class=${environment.build.hibernate.connection.driver_class}
hibernate.connection.url=${environment.build.hibernate.connection.url}
hibernate.connection.username=${environment.build.hibernate.connection.username}
hibernate.connection.password=${environment.build.hibernate.connection.password}
```

- This approach makes it easy to use different settings for different environments

uPortal Import/Export

- Provides create/update/delete (CRUD) operations on most portal entities
- Some **things have changed** since uPortal 3
- Imports are backwards-compatible to 2.5
- uPortal 4 Supports the following entities:
 - Entity Types
 - Structures
 - Themes
 - Users (template/normal)
 - Groups
 - GroupMemberships (hybrid)
 - Portlet Types
 - Portlets
 - Memberships
 - Permissions
 - Permission Owners
 - Permission Sets
 - Profiles
 - Layouts (template/normal)
 - DLM Fragment Definitions

File Types

- Import/Export uses the following file extensions:
 - .entity-type
 - .stylesheet-descriptor.xml
 - .template-user
 - .user
 - .group
 - .group_membership
 - .portlet-type
 - .portlet.xml
 - .membership
 - .permission
 - .permission_owner
 - .permission_set
 - .profile
 - .fragment-layout
 - .layout
 - .fragment-definition
 - .batch
- Portal entities are **imported in this sequence** to deal with dependencies

Example: guest-lo.fragment-layout

```
<layout xmlns:dlm="http://www.uportal.org/layout/dlm" script="classpath://org/jasig/portal/io/import-layout_v3-2.crn"
  <folder ID="s1" hidden="false" immutable="false" name="Root folder" type="root" unremovable="true">
    <folder ID="s2" hidden="true" immutable="true" name="Header folder" type="header" unremovable="true">
      <channel fname="fragment-admin-exit" unremovable="false" hidden="false" immutable="false" ID="n3"/>
    </folder>
    <folder ID="s4" hidden="false" immutable="false" name="Welcome" type="regular" unremovable="false">
      <folder ID="s5" hidden="false" immutable="false" name="Column" type="regular" unremovable="false">
        <structure-attribute>
          <name>width</name>
          <value>60%</value>
        </structure-attribute>
        <channel fname="what-is-uportal" unremovable="false" hidden="false" immutable="false" ID="n6"/>
        <channel fname="logging-in" unremovable="false" hidden="false" immutable="false" ID="n7"/>
      </folder>
      <folder ID="s8" hidden="false" immutable="false" name="Column" type="regular" unremovable="false">
        <structure-attribute>
          <name>width</name>
          <value>40%</value>
        </structure-attribute>
        <channel fname="google-portlet" unremovable="false" hidden="false" immutable="false" ID="n9"/>
        <channel fname="uportal-links" unremovable="false" hidden="false" immutable="false" ID="n10"/>
        <channel fname="please-register" unremovable="false" hidden="false" immutable="false" ID="n11"/>
      </folder>
    </folder>
    <folder ID="s12" hidden="false" immutable="false" name="Footer folder" type="footer" unremovable="false"/>
  </folder>
</layout>
```

Example: student.user

```
<user script="classpath://org/jasig/portal/io/import-user_v3-2.crn" username="student">  
  <default-user>defaultTemplateUser</default-user>  
  <person-directory>  
    <encrptd-pswd>(MD5)mhmjKvf2F3gPizS9DrA+CsFmqj74oTSb</encrptd-pswd>  
  </person-directory>  
</user>
```








Using Import/Export

- Ant tasks:
 - **crn-export**: Creates XML representations of the specified object(s)
 - **crn-import**: Modifies the database to match the specified XML document(s)
 - **crn-delete**: Removes the specified object(s) from the database

```
>ant crn-export -Ddir=mylayouts -Dtype=all-layouts
```

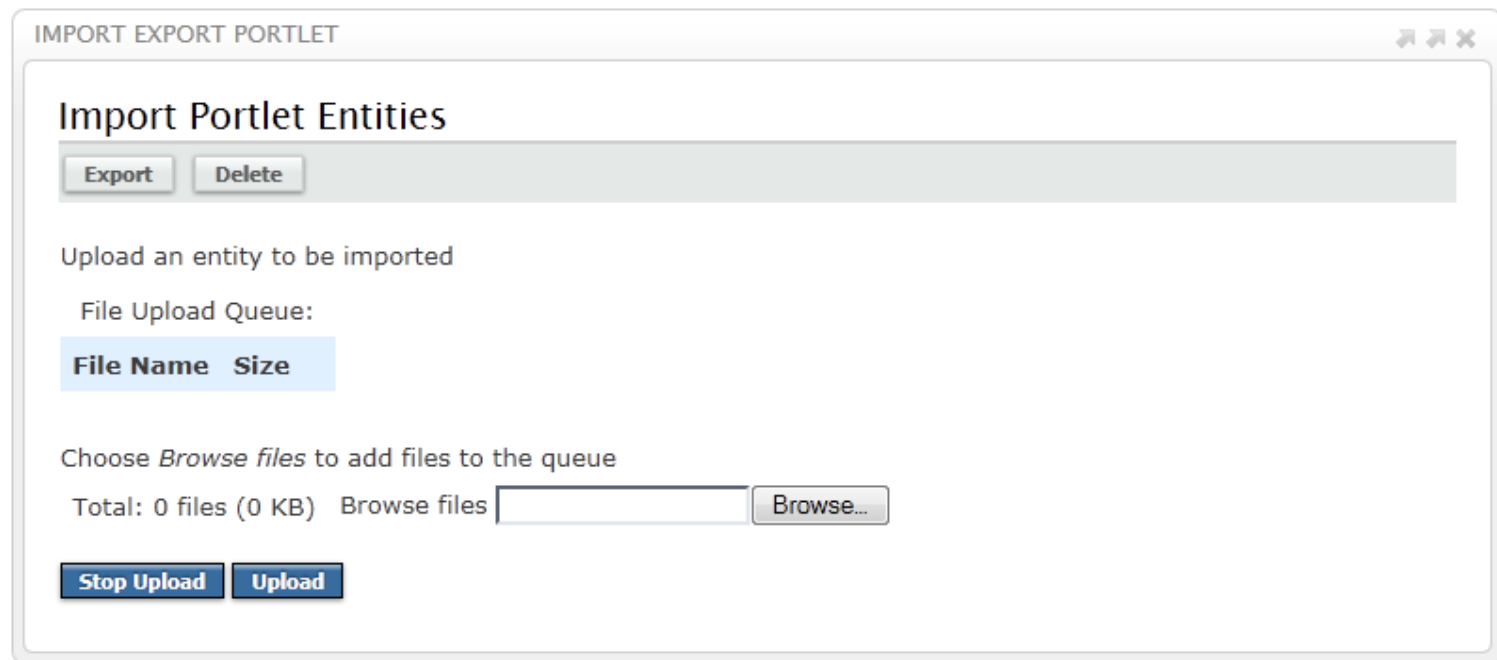
uPortal Manual Documentation

- Check the uPortal Manual for a comprehensive list of supported types and usage notes

entity-type	sysid	export	delete	notes
all		All portal entities		Combines the following sub-types: <ul style="list-style-type: none">• all-layouts• all-profiles• all-permission_sets• all-channels• all-channel-types• all-users• all-themes• all-structures• all-entity-types• all-group_memberships• all-fragment-definitions
all-layouts		All user layouts		Normal users generate <code>.layout</code> files on export; fragment owners generate <code>.fragment-layout</code>
all-profiles		All user profiles		May generate more than one document per user

ImportExport Portlet

- Provides access to uPortal Import/Export capabilities from the portal UI



IMPORT EXPORT PORTLET

Import Portlet Entities

Export Delete

Upload an entity to be imported

File Upload Queue:

File Name	Size
-----------	------

Choose *Browse files* to add files to the queue

Total: 0 files (0 KB) Browse files Browse...

Stop Upload Upload

- You can restrict allowable operations at deploy/publish time with Portlet Preferences

ImportExport Portlet (cont.)

- Import Operations: any valid document
- Export Operations:

```
layout | portlet | portlet-type | group | user  
| theme | structure | entity-type |  
fragment-definition
```

- Delete Operations (use *extreme caution*):

```
entity-type | portlet | portlet-type | group |  
layout | structure | theme user | fragment-  
definition
```

entities.location

- You can change the location of entity files in build.properties
- Advantages:
 - You can replace your whole universe of data with one property change
 - You can use different data sets for different environments

>ant initdb

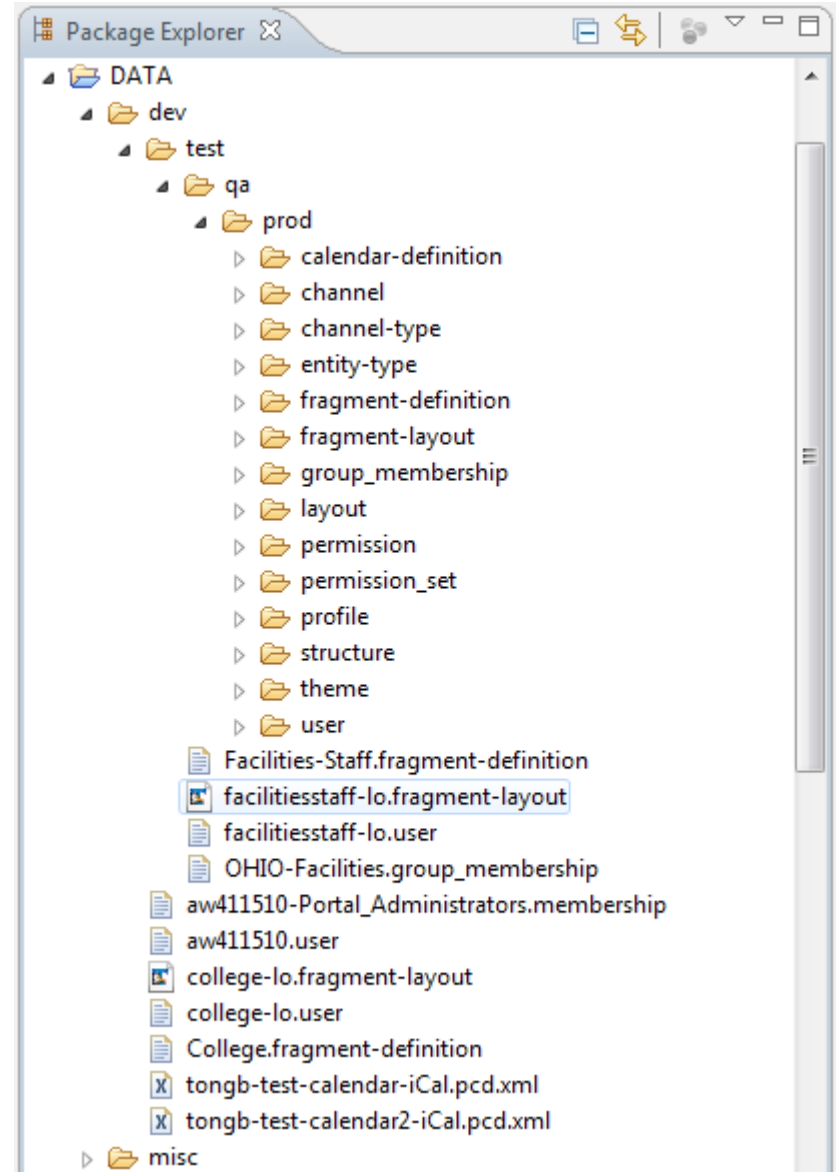
- You can even do...

>ant initdb -Dentities.location=*myDir*

- You can maintain both your data and uPortal sample data in Source Code Control

Hierarchical Data Directories

- Use a nested series of directories named for each environment
- Use entities.location in each build.properties to match environments with directories
- Promote data from one environment to another by moving it down the chain
- **WARNING:** Doesn't work with multiple versions of the same entity



Larger Upgrades

*Some notes on upgrades
bigger than patch releases*

Vendor Branching

- It **may** be possible to use a vendor branch to change minor releases (e.g. 3.1.x → 3.2.x)
- It **won't** be possible to use a vendor branch to change major releases (e.g. 3.x → 4.x)
- Files tend to move and/or evolve in major and minor releases
- If you choose vendor branching, consider creating a new one in either case

Configuration & Customization

- Catalog local changes to the portal
- *The fewer there are, the more feasible* it will be to upgrade
- Re-apply configuration manually – files move and names of settings change between versions
- Often feature enhancements & bug fixes need not be re-applied because they've been added to the project itself (though not always in identical form)

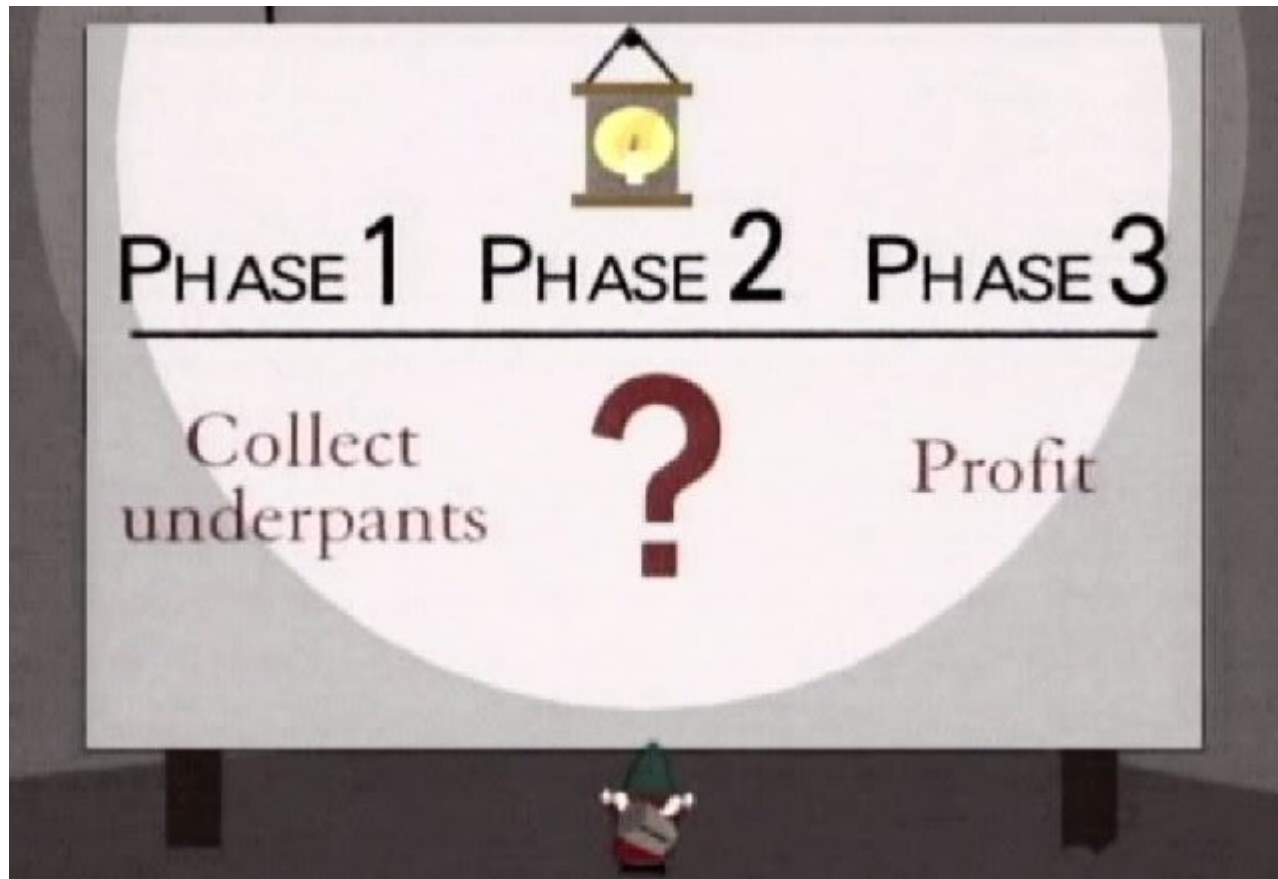
Theme & Skin

- Much of the coolness of uP3 and uP4 is baked-into the **Universality** theme
- So it's best *not* to try porting an earlier theme or skin to a later version of the portal
- This is another area for a do-over

Data Migration

- Database schema changes can & do happen with both major and minor releases (but not patch)
- That means you can't use an earlier database with a more recent version of the portal
- Thankfully there is a simple, 3-step procedure...

Three Step Procedure



The Real Three Step Procedure

1. Export your existing data
2. Make *appropriate* modifications*
3. Import your data to the new portal database

**In other words, '?'*

1. Export

- Consider specifying an `included_users_file` in `export.properties`
- Allows you to *exclude* data for non-listed users pertaining to:
 - user accounts
 - layouts
 - group memberships
 - permissions
- Fragment owners are included anyway, even if they aren't listed
- Useful as a “reset” button if your fragments & portlets are changing dramatically

2. Make Appropriate Changes

- Sometimes the example uPortal data makes a change you want to make as well
- Sometimes moving to a new framework version means you want to do something differently
- You may also want to clean up data that is no longer relevant
- This process is somewhat *subjective*, and different for everyone

3. Import

- Use `import.properties` to make the following changes across all users, if desired:
 - Structure
 - Theme
 - Template User

Final Thoughts

- Do a few test runs – review the results and refine Step #2
- Consider scripting Step #2 if you need to preserve up-to-the-minute data changes
- If desired, you can migrate one node at a time for zero downtime
- An upgrade is a great time to *clean house* – consider contributing your bug fixes & innovations!

Questions?



Cris Holdorph

holdorph@unicon.net

Drew Wills

drew@unicon.net